Dissimilarity Plots: A Visual Exploration Tool for Partitional Clustering

Michael Hahsler
Department of Computer Science and Engineering, Lyle School of Engineering,
Southern Methodist University, Dallas, TX, USA
and
Kurt Hornik
Department of Finance, Accounting and Statistics,
Wirtschaftsuniversität Wien, Vienna, Austria

August 8, 2010

Abstract

For hierarchical clustering, dendrograms are a convenient and powerful visualization technique. Although many visualization methods have been suggested for partitional clustering, their usefulness deteriorates quickly with increasing dimensionality of the data and/or they fail to represent structure between and within clusters simultaneously. In this paper we extend (dissimilarity) matrix shading with several reordering steps based on seriation techniques. Both ideas, matrix shading and reordering, have been well-known for a long time. However, only recent algorithmic improvements allow us to solve or approximately solve the seriation problem efficiently for larger problems. Furthermore, seriation techniques are used in a novel stepwise process (within each cluster and between clusters) which leads to a visualization technique that is able to present the structure between clusters and the micro-structure within clusters in one concise plot. This not only allows us to judge cluster quality but also makes mis-specification of the number of clusters apparent. We give a detailed discussion of the construction of dissimilarity plots and demonstrate their usefulness with several examples. Experiments show that dissimilarity plots scale very well with increasing data dimensionality.

Supplemental materials with additional experiments for this paper are available online.

Key Words: Visualization; Clustering; Matrix shading; Seriation.

1 Introduction

Assessing the quality of an obtained cluster solution has been a research topic since the invention of cluster analysis. This is especially important since all popular clustering algorithms produce a clustering even for data without a "cluster" structure. The quality of clustering or individual clusters is typically judged by intra and inter-cluster similarities. High intra-cluster similarity (between objects in the same cluster) and at the same time low inter-cluster similarity (between different clusters) indicate a good clustering. To present these similarities visualizations are helpful for judging the quality of a clustering and to explore the cluster structure.

For hierarchical clustering, dendrograms (Hartigan, 1967) are available which show the hierarchical structure of the clustering as a binary tree. Similarities between clusters and between objects are represented in the plot by the height of internal nodes of the tree. Cluster quality can be judged by looking at the distance between the internal nodes that separate clusters and the nodes that separate the objects in each cluster. Unfortunately such a convenient visualization is only possible for hierarchical/nested clusterings.

For an arbitrary partition of data into k clusters, the original objects can be projected into 2-dimensional space using dimensionality reduction methods (e.g., principal component analysis or multi-dimensional scaling). Objects belonging to the same cluster can be marked and separation between clusters can be judged visually (Pison et al., 1999). This type of visualization works well only if the dimensionality reduction method preserves a large portion of the dimensions which separate the clusters. The usefulness of the display deteriorates if the separating dimension has a relatively small variability compared to the other dimensions which becomes more likely as the dimensionality of the data increases.

Another approach is to visualize metrics calculated from inter and intra-cluster similarities. For example, silhouette width (Rousseeuw, 1987; Kaufman and Rousseeuw, 1990) is a measure for how much an object belongs to its cluster (intra cluster similarity) compared to how close it is to objects in its nearest neighboring clusters. Silhouette widths are typically visualized using a barplot where the objects are ordered by cluster and decreasing silhouette width. However, this type of visualization only provides a diagnostic tool for cluster quality and does not allow analysis of the structure of the data. These and several other visualization methods (e.g., based on self-organizing maps and neighborhood graphs) are reviewed in Leisch (2008).

The visualization technique presented in this paper is based on a different technique called matrix shading (see, e.g., Sneath and Sokal, 1973; Ling, 1973; Gale et al., 1984; Huband et al., 2005). For matrix shading, each value in the matrix is represented by a square with the intensity of the color depending on the value. The presentation is improved by reordering the rows and columns. Reordering matrices is a long known technique. For example, Jacques Bertin devotes a whole chapter of his book "Graphics and Graphic Information Processing" (Bertin, 1981, which was first published in French in 1967) to this topic. More recently matrix reordering was applied to mosaic displays for multi-way contingency tables (Friendly, 1994), distance matrices (Wishart, 1999), correlation matrices (Friendly, 2002), and scatter plot matrices (Hurley, 2004). For these applications reordering is typically done using heuristics. For example matrix shading is often used in connection with hierarchical clustering, where the order of the dendrogram leaf nodes is used to arrange the matrix yielding a cluster heat map (Eisen et al., 1998; Wilkinson and Friendly, 2009). Since the order of leaf nodes in a dendrogram is not unique (each subtree can be rotated) and to further improve the presentation, the leaf nodes can be reordered using heuristics (e.g., Gruvaeus and Wainer, 1972). Only more recently Bar-Joseph et al. (2001) developed an $O(n^4)$ algorithm that finds the optimal order of leaf nodes which minimizes the sum of distances between the nodes in the order.

The idea to apply matrix shading not only with hierarchical clustering but also in the context of partitional clustering is obvious and is used in a method called *CLUSION* suggested by Strehl and Ghosh (2003) for a graph-based partitional clustering. In this method the dissimilarity matrix is arranged such that all objects pertaining to a single cluster appear in consecutive order in the matrix. The authors call this *coarse seriation*. The result of a "good" clustering should be a matrix with low dissimilarity values forming blocks around the main diagonal corresponding to the clusters. However, using coarse seriation, the order of the clusters is only an artifact of the cluster algorithm and the objects within each cluster are unordered potentially obscuring structure in the data.

Seriation is the combinatorial problem of arranging a set of objects in a linear order given available data and some loss function. The dissimilarity plot method which we first briefly introduced in Hahsler et al. (2008) and describe in this paper in detail applies state-of-the-art seriation methods to find, given a partitional clustering, the optimal or near optimal positions of clusters and objects within clusters in a shaded dissimilarity matrix. It aims at visualizing global structure (similarity between different clusters is reflected by their position relative to each other) as well as the micro structure within each cluster (position of objects) to be able to judge cluster quality and give an indication whether the number of clusters was mis-specified. Seriation is a combinatorial problem and thus in general very difficult to solve for all but extremely small problems. Recently a very efficient algorithm for the seriation problem based on branch-and-bound (Brusco and Stahl, 2005) and a heuristic that combines dynamic programming combined with simulated annealing (Brusco et al., 2008) have been developed. This algorithmic progress and the fact that we only need to apply seriation to subsets of the data allow us to use seriation techniques for the visualization of partitional clusterings of larger data sets.

The rest of the paper is organized as follows. In Section 2 we introduce the seriation problem and its application for optimally positioning clusters and objects in the plot. The method used to produce dissimilarity plots is described in Section 3. Sections 4 and 5 present several examples and experiments to show how dissimilarity plots compare to other methods. We conclude the paper with Section 6.

2 Seriation

Seriation is a basic problem in combinatorial data analysis (Arabie and Hubert, 1996) with the aim to arrange all objects in a set in a linear order given available data and some loss function, in order to reveal structural information. Solving problems in combinatorial data analysis requires the solution of discrete optimization problems which, in the most general case, involves evaluating all feasible solutions. Due to the combinatorial nature, the number of possible solutions grows with problem size (number of objects, n) at the order $O(n!)$. This makes a brute-force enumerative approach infeasible for all but very small problems. To solve larger problems (currently with up to 40 objects), partial enumeration methods can be used. For example, Hubert et al. (1987) propose dynamic programming and Brusco and Stahl (2005) use a branch-and-bound strategy. For larger problems heuristics can be employed. Recently a heuristic which combines dynamic programming with simulated annealing was developed by Brusco et al. (2008). This heuristic typically produces very good average results and is able to find close to optimal solutions for much larger problems.

To seriate a set of n objects $\mathcal{O} = \{O_1, \dots, O_n\}$ one typically starts with an $n \times n$ symmetric dissimilarity matrix $\mathbf{D} = (d_{ij})$ where d_{ij} for $1 \leq i, j \leq n$ represents the dissimilarity between objects O_i and O_j, and $d_{ii} = 0$ for all i. We define a permutation function Ψ as a function which reorders the objects in $\mathbf{D}$ by simultaneously permuting rows and columns, i.e., $\Psi(\mathbf{D}) = \mathbf{ADA}^{\top}$, where $\mathbf{A}$ is a permutation matrix with exactly one 1 per row and column and otherwise only 0s. The permutation matrix $\mathbf{A}$ can be obtained by permuting the rows of an $n \times n$ identity matrix according to the order the objects in $\mathcal{O}$ should have in the permutation.

The seriation problem is to find a permutation function Ψ^* which optimizes the value of a given loss function L. This results in the optimization problem

$$\Psi^* = \underset{\Psi}{\operatorname{argmin}}\, L(\Psi(\mathbf{D})). \tag{1}$$

Next, we introduce a small selection of loss functions which can be used for ordering objects and clusters in dissimilarity plot. A more comprehensive list of loss functions used for the seriation problem can be found in Hahsler et al. (2008).

2.1 Column/row gradient measures

A symmetric dissimilarity matrix where the values in all rows and columns do not decrease when moving away from the main diagonal is called a perfect *anti-Robinson matrix* after the statistician Robinson (1951). Formally, an $n \times n$ dissimilarity matrix $\mathbf{D}$ is in anti-Robinson form if and only if the following two gradient conditions hold (Hubert et al., 1987):

$$\text{within rows:} \quad d_{ik} \leq d_{ij} \quad \text{for} \quad 1 \leq i < k < j \leq n; \tag{2}$$

$$\text{within columns:} \quad d_{kj} \leq d_{ij} \quad \text{for} \quad 1 \leq i < k < j \leq n. \tag{3}$$

In an anti-Robinson matrix the smallest dissimilarity values appear close to the main diagonal, therefore, the closer objects are together in the order of the matrix, the higher their similarity. This provides a natural objective for the seriation problem.

It has to be noted that $\mathbf{D}$ can be brought into a perfect anti-Robinson form by row and column permutation whenever $\mathbf{D}$ is an ultrametric or $\mathbf{D}$ has an exact Euclidean representation in a single dimension (Hubert et al., 1987). However, for most data only an approximation to the anti-Robinson form is possible.

A suitable loss measure which quantifies the divergence of a matrix from the anti-Robinson form was given by Hubert et al. (1987) as

$$L(\mathbf{D}) = \sum_{i<k<j} f(d_{ik}, d_{ij}) + \sum_{i<k<j} f(d_{kj}, d_{ij}) \tag{4}$$

where $f(\cdot, \cdot)$ is a function which defines how a violation or satisfaction of a gradient condition for an object triple (O_i, O_k and O_j) is counted. Hubert et al. (1987) suggest two functions. The first function is given by:

$$f(y, z) = \operatorname{sign}(y - z) = \begin{cases} -1 & \text{if} \quad z > y; \\ 0 & \text{if} \quad z = y; \\ +1 & \text{if} \quad z < y. \end{cases} \tag{5}$$

It results in the raw number of triples violating the gradient constraints minus triples which satisfy the constraints.

The second function is defined as:

$$f(y, z) = |y - z| \operatorname{sign}(y - z) = y - z \tag{6}$$

It weighs each satisfaction or violation by its magnitude given by the absolute difference between the values.

2.2 Anti-Robinson events

An even simpler loss function can be created in the same way as the gradient measures above by concentrating on violations only.

$$L(\mathbf{D}) = \sum_{i<k<j} f(d_{ik}, d_{ij}) + \sum_{i<k<j} f(d_{kj}, d_{ij}) \tag{7}$$

To only count the violations we use

$$f(y, z) = I(y, z) = \begin{cases} 1 & \text{if} \quad z < y \quad \text{and} \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

Chen (2002) presented a formulation for an equivalent loss function and called the violations *anti-Robinson events*. The maximal number of anti-Robinson events is given by $(n-2)(n-1)n/3$ and occurs if the matrix is in perfect *Robinson form* (values do not increase when moving away from the main diagonal).

Chen (2002) also introduced a weighted versions of the loss function resulting in

$$f(y, z) = |y - z| I(y, z) \tag{9}$$

using the absolute deviations as weights.

2.3 Hamiltonian path length

The dissimilarity matrix $\mathbf{D}$ can be represented as a finite weighted graph $G = (\Omega, E)$ where the set of objects Ω constitute the vertices and each edge $e_{ij} \in E$ between the objects $O_i, O_j \in \Omega$ has a weight w_{ij} associated which represents the dissimilarity d_{ij}.

Such a graph can be used for solving a seriation problem (see, e.g., Hubert, 1974; Caraux and Pinloche, 2005). An order Ψ of the objects can be seen as a path through the graph where each node is visited exactly once, i.e., a Hamiltonian path. Minimizing the Hamiltonian path length results in an order optimal with respect to minimizing the dissimilarities between neighboring objects. The loss function based on the Hamiltonian path length is:

$$L(\mathbf{D}) = \sum_{i=1}^{n-1} d_{i,i+1}. \tag{10}$$

The length of the shortest Hamiltonian path is equal to the value of the *minimal span loss function* (as used by Chen, 2002), and finding the associated order is equivalent to solving the *traveling salesperson problem (TSP)* (Gutin and Punnen, 2002). For the TSP exist specialized solvers (e.g., Concorde by Applegate et al. (2006)) and good heuristics (e.g., Lin and Kernighan, 1973) which are more efficient than algorithms which try to approximate the anti-Robinson forms.

3 Dissimilarity plots

The aim of visualizing a clustering solution is to help to make the structure or lack thereof implied by the cluster solution apparent. To achieve this goal we use matrix shading to display the dissimilarity matrix with the following improvements:

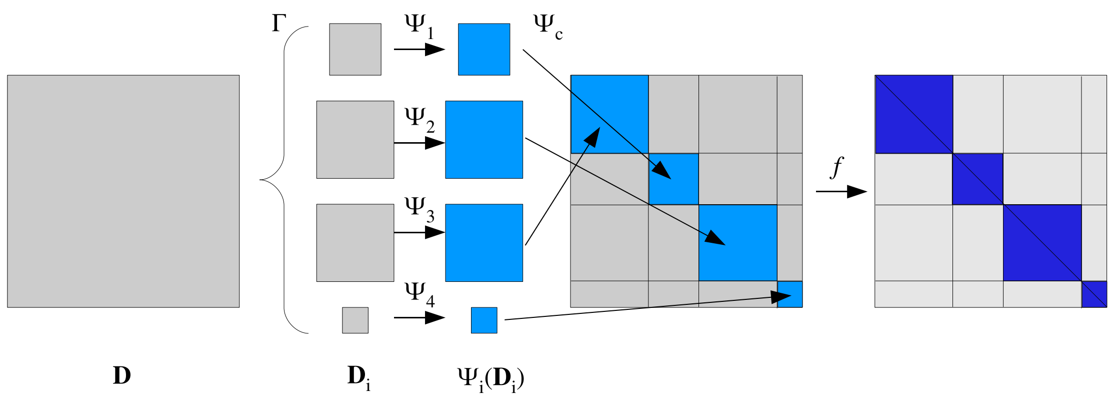


Figure 1: Example of the application of reordering for dissimilarity plot with four clusters.

1. We arrange the clusters in such a way that clusters which contain objects that are more similar are displayed closer together. This helps to understand the relationship between clusters. For example, if the clustering algorithm breaks apart a natural group in the data, the two clusters formed should be displayed next to each other.

2. We arrange the objects in each cluster such that more similar objects are displayed closer together. This way the micro-structure inside each cluster can be analyzed. For example a cluster can contain two or more quite distinct groups which might indicate that the number of clusters used for clustering was too small.

3. We use a color scheme that is equivalent to a monotone, possibly non-linear transformation of the (dis)similarity data to highlight different aspects of the clustering (e.g., cluster compactness or inter cluster similarities).

Figure 1 shows the steps needed for visualizing data using dissimilarity plots. As input we have a dissimilarity matrix $\mathbf{D}$ and the assignment function $\Gamma : \mathcal{O} \to \{1, \dots, k\}$ provided by a partitional clustering algorithm, which assigns a cluster identification number to each object. This function is used to split the set of objects $\mathcal{O}$ into k subsets:

$$\mathcal{O}_i = \{o \in \mathcal{O} \mid \Gamma(o) = i\} \quad \text{for } i = 1 \dots k. \tag{11}$$

We create for each subset $\mathcal{O}_i$ representing cluster i a sub-matrix $\mathbf{D}_i$ containing only the dissimilarities between the objects in $\mathcal{O}_i$.

To reveal structural information within each cluster, we use a seriation method on the set of objects for each cluster using the corresponding dissimilarity sub-matrix $\mathbf{D}_i$ resulting in k permutation functions Ψ_i which are used to permute the columns and rows of $\mathbf{D}_i$ representing objects in each cluster. Note that we never have to apply the typically expensive seriation method to the whole dissimilarity matrix.

Next, we determine the order of clusters for display. To position the clusters in the dissimilarity plot a seriation method is applied to an inter-cluster dissimilarity matrix $\mathbf{D}_c$ to find a cluster permutation function Ψ_c which determines the order of clusters in the plot. To construct $\mathbf{D}_c$ we have to compute aggregate dissimilarities between all pairs of clusters given dissimilarities between all elements of the clusters in $\mathbf{D}$. For hierarchical clustering dissimilarities between two clusters represented by two sets of objects $\mathcal{X}$ and $\mathcal{Y}$ are typically calculated by one of the following methods.

$$\text{complete-link:} \quad d_c(\mathcal{X},\mathcal{Y}) = \max\{d(x,y) : x \in \mathcal{X}, y \in \mathcal{Y}\} \tag{12}$$

$$\text{single-link:} \quad d_s(\mathcal{X},\mathcal{Y}) = \min\{d(x,y) : x \in \mathcal{X}, y \in \mathcal{Y}\} \tag{13}$$

$$\text{average-link:} \quad d_a(\mathcal{X},\mathcal{Y}) = \frac{1}{|\mathcal{X}| \cdot |\mathcal{Y}|} \sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} d(x,y) \tag{14}$$

Complete-link (single-link) respectively use the largest (smallest) possible dissimilarity between any two objects, one of each set. Average-link computes the average of all pairwise dissimilarities between objects of the two sets. In set theory the *Hausdorff metric* (Hausdorff, 2001) is used to calculate the dissimilarity between two sets defined from pairwise dissimilarities between the elements in the two set. This metric is defined as

$$d_H(\mathcal{X},\mathcal{Y}) = \max\{\sup_{x\in\mathcal{X}} \inf_{y\in\mathcal{Y}} d(x,y), \sup_{y\in\mathcal{Y}} \inf_{x\in\mathcal{X}} d(x,y)\} \tag{15}$$

The Hausdorff metric pairs up each element from one set with the most similar element from the other set and then finds the largest dissimilarity in such element pairs.

The last step in Figure 1 deals with the color palette employed. Although this is an important step, color palettes are rarely discussed for matrix shading. Using a perceptually-based color space like the CIELUV or the HCL color space (see Zeileis et al., 2009), values in the matrix can be rendered to be perceived to go linearly from a very dark gray (or black) for very low dissimilarity to a very light gray (or white) for high dissimilarity. This somewhat non-intuitive convention to use black for low values and white for high values can be interpreted as visualizing similarities instead of dissimilarities. Dissimilarities can always be transformed to similarities. However, this transformation is typically non-linear since it maps dissimilarity values with possibly unbounded domain $[0, \infty)$ to similarity values in $[0, 1]$. This creates the problem that if dissimilarities are shown in a linear gray scale then similarities are not displayed linearly and vice versa.

A solution is to abandon the idea of a linear scale and focus on the intended purpose of the visualization, e.g., on the compactness of clusters, on the micro structure within clusters, or on the interaction between clusters. This can be done by using monotone and possibly non-linear transformations of the dissimilarity values to the gray values used for visualization. Here we assume that we have an output device which can display points in a gray scale going from 0 (white or a light gray) to 1 (black or a dark gray) which is perceived to be linear by the user (Zeileis et al., 2009). We define the transformation function as

$$f : [0, d_{max}] \to [0, 1], \tag{16}$$

where d_{max} is some maximal dissimilarity value, such that for all $d \in \mathbf{D}$ we have $d \leq d_{max}$.

Some examples for useful transformations are presented in Table 1 and depicted in Figure 2. Next to the linear transformation (a) simple non-linear transformations ((b) and (c)) can be used. (b) focuses the attention on compact clusters and reduces high dissimilarity "noise" while (c) makes even areas of higher dissimilarity visible. Note that the linear transformation (a) is just a special case of the non-linear transformation with $p = 1$. (d) and (e) show two possible transformations that highlight areas of lower dissimilarity. (d) simply cuts the palette off after the threshold while (e) uses a smooth cutoff based on the Logistic cumulative distribution function.

Since we deal with symmetric dissimilarity matrices, the display is mirrored around the diagonal which is redundant. We can use the lower triangle of the plot to display the aggregated dissimilarities within and between clusters already calculated for the inter-cluster dissimilarity matrix $\mathbf{D}_c$.

The usefulness of a display like a dissimilarity plot has to be judged by its ability to present the cluster structure in a clear form to the user. In the following sections we will present several examples and experiments to show the usefulness of dissimilarity plots.

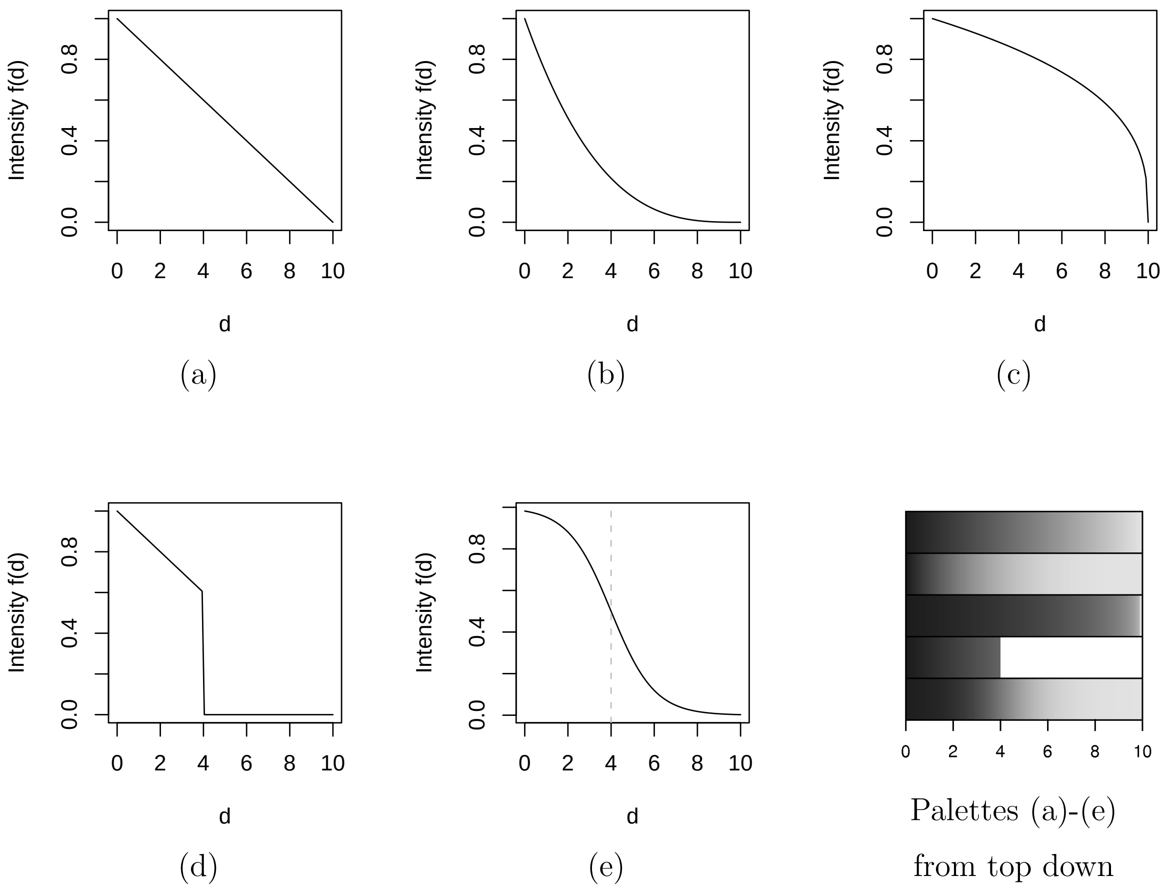


Figure 2: Transformation functions from Table 1 with $d_{max} = 10$: (a) linear palette, (b) sub-linear with $p = 3$, (c) super-linear with $p = 1/3$, (d) linear with threshold at $t = 4$, and (e) non-linear (based on a logistic cumulative distribution function) with $t = 4$ and $s = 1$.

Figure 2	Description	Transformation functions
(a)	Linear palette	$f(d) = 1 - d/d_{max}$
(b), (c)	Non-linear palette using the power p	$f(d) = (1 - d/d_{max})^p$
(d)	Linear palette with threshold t	$f(d) = \begin{cases} 1 - d/d_{max} & \text{if } d \leq t, \\ 0 & \text{otherwise.} \end{cases}$
(e)	Non-linear palette (based on the Logistic cumulative distribution function) with a smooth cutoff at the location parameter t and with scale parameter s	$f(d) = 1 - \frac{1}{(1+\exp((d-t)/s))}$

Table 1: Examples of transformation functions.

4 Examples

In this section we present several examples to show the potential of dissimilarity plots. For the examples we use the column/row gradient measure as the loss function for the seriation method and aggregate dissimilarities between clusters using average pairwise dissimilarity. The reordering of clusters is done using branch-and-bound to find the optimal solution (Brusco and Stahl, 2005). For the reordering for all objects in each cluster we use a simulated annealing heuristic (Brusco et al., 2008). The implementation of both algorithms is provided by Michael Brusco and is available in the R extension package **seriation** (Hahsler et al., 2010).

4.1 Easily distinguishable groups

First we look at the *Ruspini* data set (Ruspini, 1970) which is popular for illustrating clustering techniques. It consists of 75 points in two-dimensional space with four clearly distinguishable groups.

We calculated a dissimilarity matrix using the euclidean distance and we used a k-medoids clustering algorithm (partitioning around medoids (PAM); Kaufman and Rousseeuw, 1990) to produce a partition with $k = 4$ clusters. We present several visualizations of the clustering in Figure 3. Figure 3(a) plots the data as a scatter plot with clusters annotated. The four clusters are clearly separated and it is visible that clusters 2 and 4 are the closest clusters. Figure 3(b) shows the silhouette widths for all objects. The clean clustering is represented here by the fact that all objects have large silhouette widths indicating that they fit well in the cluster they are assigned to and that they are far from the objects in the next closest cluster. Figure 3(c) shows the dissimilarity matrix reordered using coarse seriation (Strehl and Ghosh, 2003) where all objects pertaining to a single cluster appear in consecutive order. Dissimilarity values are represented by gray values given in the color key below the plot. The clearly visible dark squares on the diagonal and the lighter off-diagonal blocks indicate that the clusters are well defined. However, the structure between the clusters and within clusters is not preserved in the plot. Figure 3(d) shows the dissimilarity plot. In addition to the clustering information already visible using coarse seriation, the position of the clusters and the average cluster dissimilarity plotted in the lower left triangle in the dissimilarity plot indicate that clusters 3 and 1 and clusters 2 and 4 are closer together while 3 and 4 (the endpoints of the plot) are the most dissimilar. For the intra cluster structure it is interesting that the darker triangle above the main diagonal for cluster 2 is not perfect. Close examination of cluster 2 in Figure 3(d) reveals that the first 5 objects in that cluster are somewhat separated from the remainder of the cluster. This can be verified in Figure 3(a).

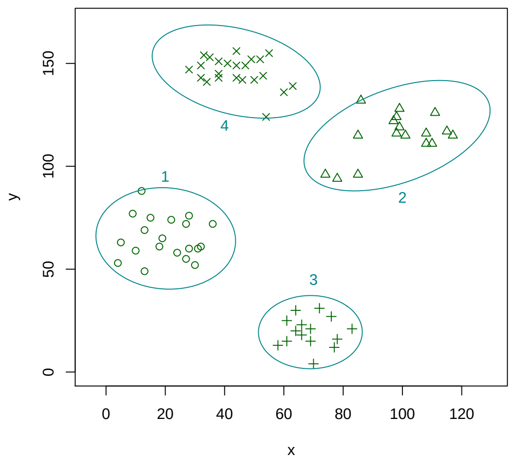
The maximum number of anti-Robinson events for the 75 points in the data set is 135050. The unordered dissimilarity matrix contains 66151 events (48.98% of the maximum). Coarse seriation produces 57452 events (42.54%) and dissimilarity plot reduces the number to 27529 events (20.38%) indicating a much clearer presentation.

For this example all techniques provided good results which was to be expected given the "easy to cluster" data set. Next we will see how dissimilarity plots help to identify a mis-specification of the number of clusters.

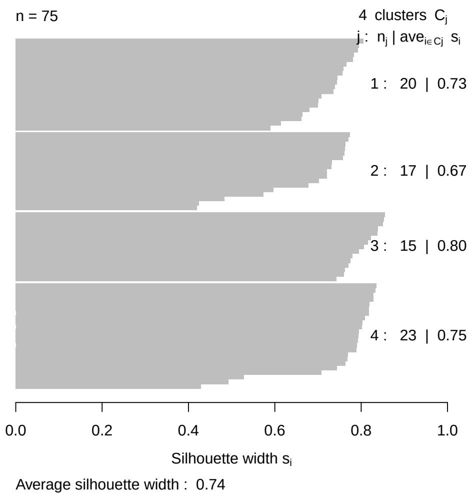
4.2 Mis-specification of the number of clusters

We again use the Ruspini data set with four groups but this time we mis-specify the number of clusters and use first $k = 3$ and then $k = 7$. Again, we use PAM to create the clusterings.

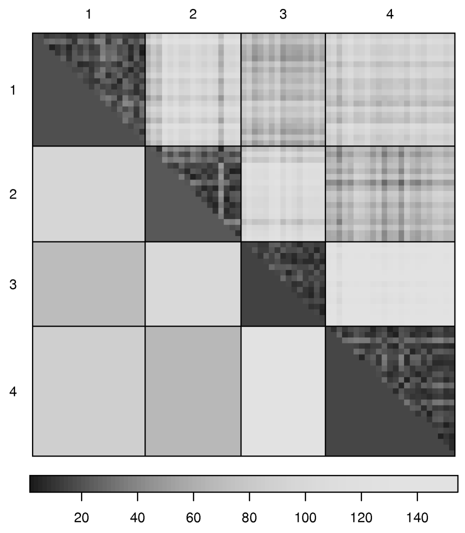
Figure 4 compares matrix shading using coarse seriation (left column) and dissimilarity plots (right column) for using three (first row) and seven (second row) clusters. In Figures 4(a) we see that clusters 2 and 3 are clearly visible as two dark squares and cluster 1 is on average less compact



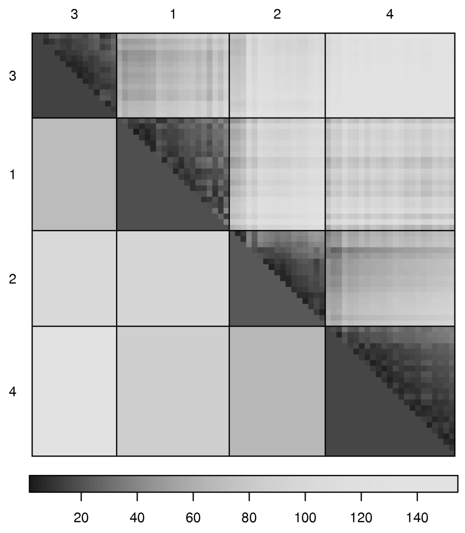
(a)



(b)

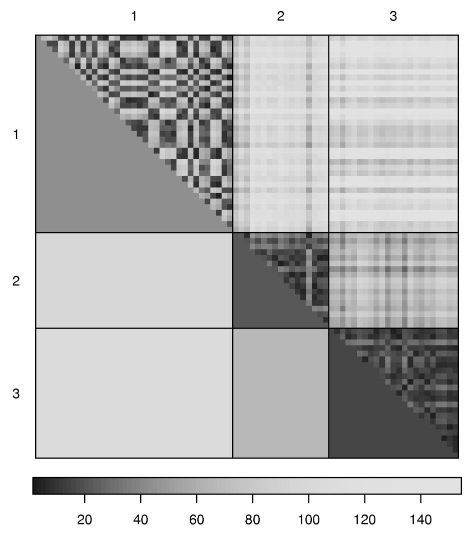


(c)

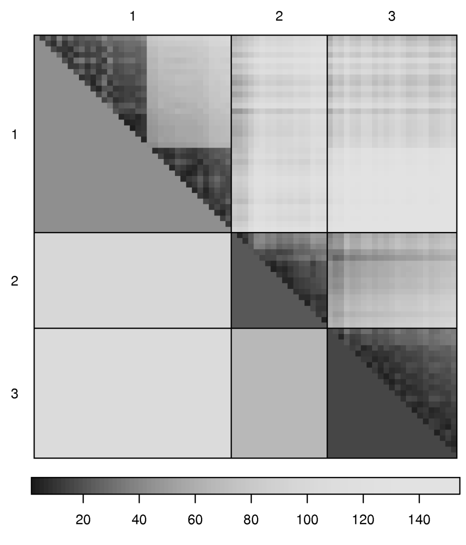


(d)

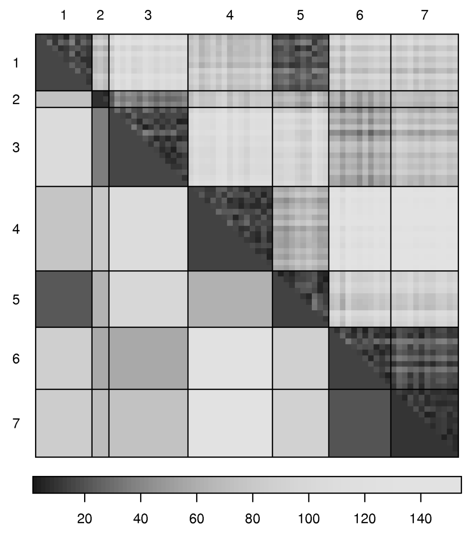
Figure 3: Plots for the Ruspini data set with 4 clearly separated clusters. (a) Scatterplot of the data, (b) silhouette plot, (c) matrix shading reordered using coarse seriation, and (d) dissimilarity plot. (c) and (d) use a non-linear palette (cubic; $p = 3$) to highlight low dissimilarities.



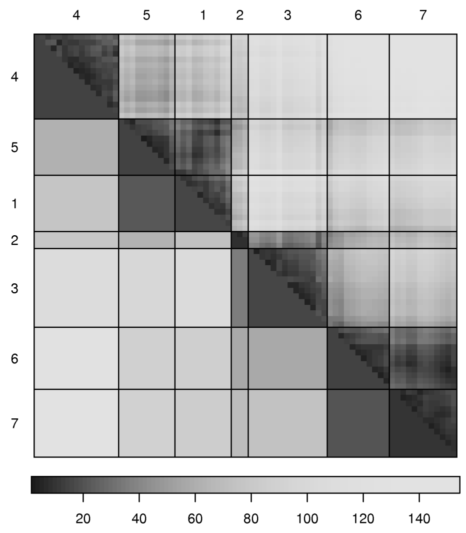
(a)



(b)



(c)



(d)

k	Maximum	Unordered		Coarse seriation		Dissimilarity plot	
3	135050	66151	(48.98%)	37156	(27.51%)	35340	(26.17%)
7	135050	66151	(48.98%)	59302	(43.91%)	22780	(16.87%)

(e)

Figure 4: Dissimilarity plots for the Ruspini data set with mis-specified number of clusters. Three instead of four clusters with (a) coarse seriation and (b) dissimilarity plot. Seven clusters with (c) coarse seriation and (d) dissimilarity plot. All plots use a non-linear palette (cubic; $p = 3$). (e) shows the number (percentage) of anti-Robinson events in the plots.

(lighter triangle below the diagonal). Looking at the relationship between the individual objects in cluster 1 (triangle above the diagonal) shows a more pronounced checkerboard structure than the other clusters. The dissimilarity plot in Figures 4(b) however is able to clearly identify two distinct groups (the two dark triangles above the diagonal). This indicates that the true number of groups must be larger than the number of clusters used for clustering. The dissimilarity plot even suggests the correct number of four clusters.

The plots in Figures 4(c) and (d) show the result for too large a number of clusters. Here the clustering algorithm breaks some natural groups into several clusters to force a partition into seven clusters. Matrix shading with coarse seriation in Figures 4(c) shows that there are several compact clusters, however since the order of clusters in the plot is just an artifact of the clustering algorithm it does not show that there are several clusters which actually should form together a larger compact cluster. The dissimilarity plot in Figure 4(d) automatically rearranges the clusters to show that some clusters belong together to form a natural group making the mis-specification of the numbers of clusters apparent.

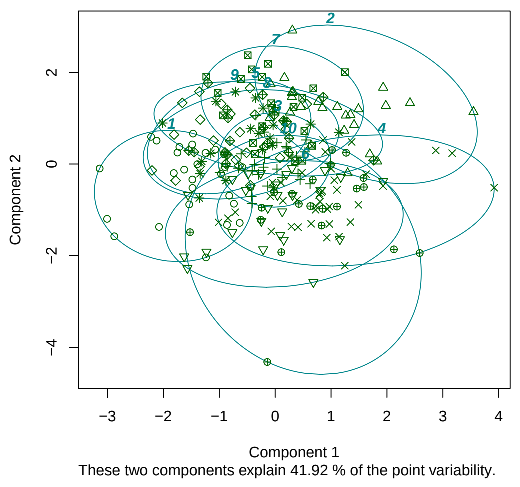
The improvement in display can also be seen in the table in Figure 4(e). Especially for $k = 7$ the number of anti-Robinson events left in the dissimilarity plot is reduced dramatically compared to coarse seriation.

4.3 Data without structure

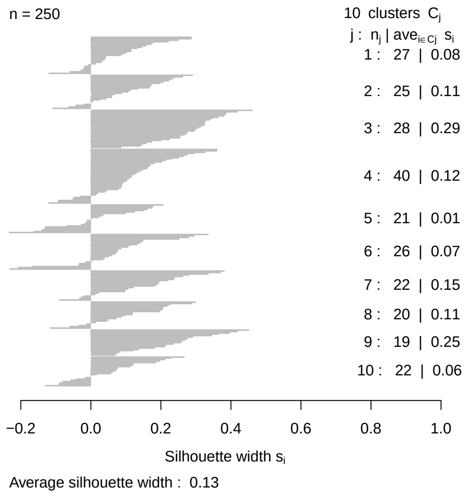
Next we look at data which do not contain any structure. Clustering algorithms will still find the specified number of clusters and it is important to identify the clustering result as an artifact rather than a “true” grouping found in the data.

We generate random data for 250 objects in five-dimensional space. The data point of each object is chosen randomly from a standard normal distribution. Distances between objects are calculated using euclidean distance and clustering is performed for $k = 10$ with PAM.

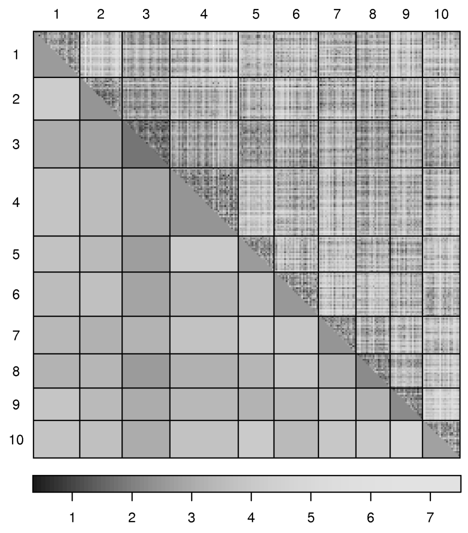
The results are presented in Figure 5. The projection of the data onto its first two principal components in Figure 5(a) shows that all clusters overlap in the plot and no structure is visible. This may indicate a “bad” clustering. However, the first two principal components only show part of the variability of the data and separation between the clusters not visible in the plot might exist. Figure 5(b) shows rather narrow silhouettes, some even with negative values which



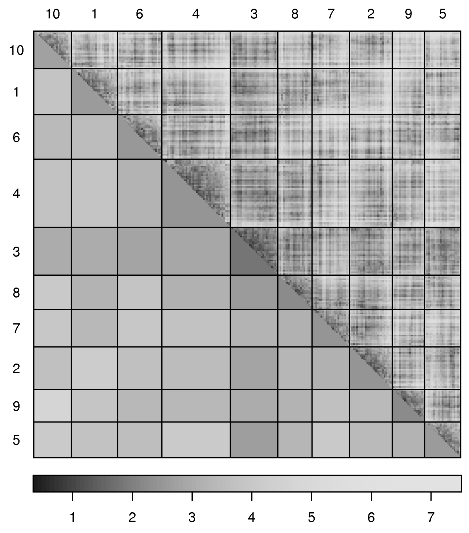
(a)



(b)



(c)



(d)

Figure 5: Plots for random data. (a) Projection of the data onto its fist two principal components, (b) silhouette plot, (c) matrix shading with coarse seriation, and (d) dissimilarity plot. Plots (c) and (d) use a non-linear palette (cubic; $p = 3$) to highlight strong clusters.

indicates that several objects in these clusters are closer to objects in other clusters than to their own medoid. Narrow silhouettes typically suggest a "weak" clustering. Both, matrix shading with coarse seriation (in Figure 5(c)) and the dissimilarity plot (in Figure 5(d)) look almost identical and show very little variation in gray value over the whole matrix. Even though we use a non-linear palette (with $p = 3$) which highlights clusters, the squares on the diagonal fade together with the rest of the plot towards a lighter gray, a clear sign that the clustering produced no useful results.

The maximum number of anti-Robinson events for the 250 objects is 5146000. The unordered dissimilarity matrix contains 2586810 events (50.27% of the maximal number). Coarse seriation reduces the number to 2395981 events (46.56%) and dissimilarity plot reduces it further to 1934304 events (37.59%). Since the data has no clustering structure, the improvement is not as large as in the prior examples. However, this actually is a desirable effect indicating again that the found clusters do not represent a useful clustering.

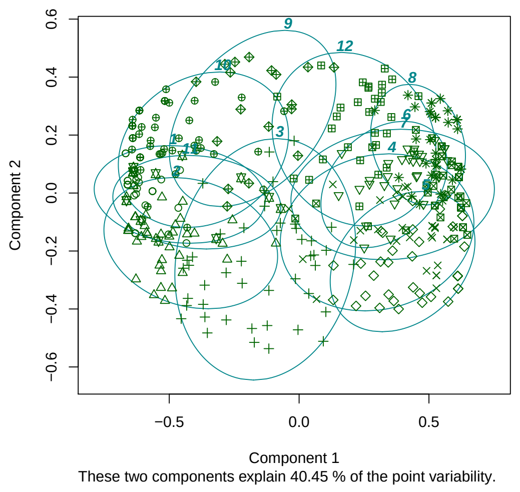
4.4 High-dimensional data

To show how dissimilarity plots work with higher-dimensional data, we use the *Votes* data set available via the UCI Repository of Machine Learning Databases (Asuncion and Newman, 2007). This data set includes votes (voted for, voted against, not voted) for each of the 435 congressmen of the U.S. House of Representatives on the 16 key votes during the second session of 1984.

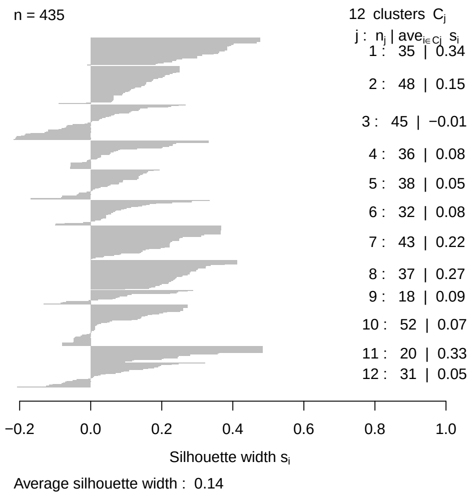
To preserve the information that some congressmen, possibly on purpose, did not vote on some topics, we encoded the data using 32 binary variables, two for each key vote. A 1 for the first variable codes for a vote in favor, a 1 for the second variable codes for a vote against, and a 0 for both indicates that the congressman did not vote on the topic. Then we used the Jaccard index (Sneath and Sokal, 1973) to calculate a dissimilarity matrix between congressmen. This is the number of votes two congressman agreed on divided by the total number of votes any of the two voted on.

For clustering again we used PAM. To decide on the number of clusters we plotted the average silhouette values for $k = 2, 3, \ldots, 30$. The first bump in the plot occurred at $k = 12$ which we selected as the number of clusters.

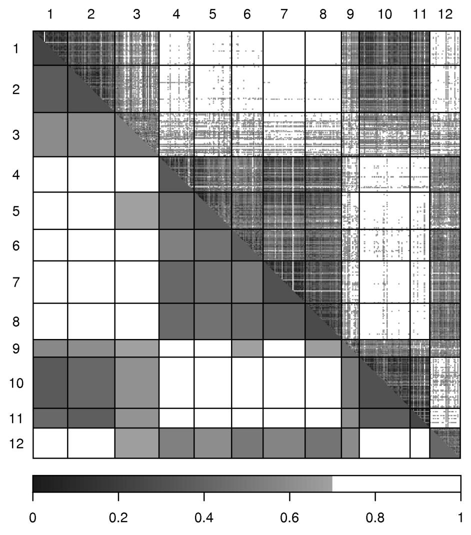
The results of the clustering are shown in Figure 6. Due to the high dimensionality, the projection of the objects onto its first two principal components in Figure 6(a) only explains 40.45%



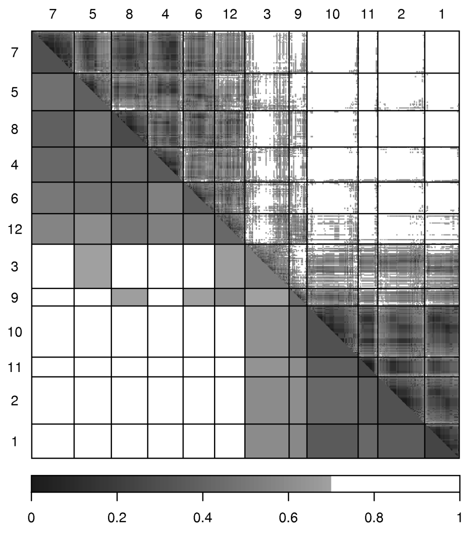
(a)



(b)



(c)



(d)

Figure 6: Plots for the Votes data set. (a) Projection of the data on its fist two principal components, (b) silhouette plot, (c) matrix shading with coarse seriation, and (d) dissimilarity plot. For (c) and (d) we use a linear palette with a shading threshold ($t = .7$) to focus on low dissimilarity areas.

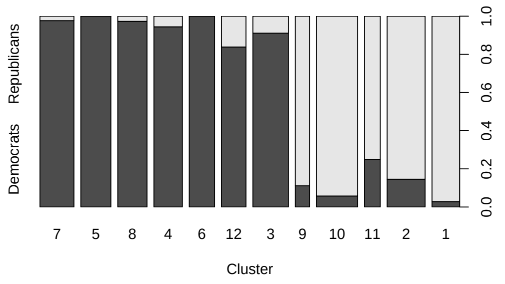


Figure 7: Visualization of cluster composition as a spine plot with reordered clusters.

of the variability and does not reveal too much apart from the observation that all the clusters seem to be arranged in a circle. The silhouette plot in Figure 6(b) shows a rather small average silhouette width of 0.14. From the individual silhouettes, it seems that cluster 3 is especially bad since most of its points are closer to points in other clusters than to its own medoid resulting in an on average negative silhouette width for the cluster. Figure 6(c) shows the result of matrix shading with coarse seriation. There are several dark blocks visible, but it is not immediately clear how they are related to each other or if there is some larger structure in the data. Figure 6(d) shows the dissimilarity plot which presents two clearly defined groups, a larger group including clusters 7, 5, 8, 4, 6 and 12 and the smaller group including clusters 10, 11, 2, 1. Two clusters (3 and 9) are related to both groups but have more similarity to the smaller group. Note that in order to make the plots clearer, we used a linear palette with threshold where we removed all dissimilarity values greater than 0.7.

The number of anti-Robinson events is reduced from 47.44% from a random order to 39.64% for coarse seriation. The dissimilarity plot further reduces the number to 21.93% indicating a large improvement.

Since the Votes data set also contains class labels (party affiliation of congressmen), we can create a cross-table for clusters and classes. The result is presented as a spine plot in Figure 7. For easier comparison, we arranged the clusters (bars) in the plot in the same order as in the dissimilarity plot. This shows that the larger block in the dissimilarity plot contains almost exclusively Democrats while the smaller cluster is dominated by Republicans. Clusters 3 and 9 consists of mostly Democrats and Republicans, respectively, who seem to share many views with Republicans but also vote on some topics with Democrats.

After analyzing the dissimilarity plot, the projection of the clusters in Figure 6(a) makes more sense. The clusters run in a half-circle from cluster 2 to cluster 5 (using Figure 7 reveals that is from Republicans to Democrats) and cluster 3 lies across all other clusters. However, this interpretation can only be reached after studying the actual class labels.

5 Experiments

In this section we study how dissimilarity plots are influenced by the number of clusters as well as the dimensionality of the data. To quantify the contribution of the reordering techniques applied for dissimilarity plots, we use the percentage of anti-Robinson events (relative to maximal possible events) left in the plot. The rationale is that a better organization of objects leads to a clearer and more useful presentation of the clustering.

For this small scale experiment we use different simulated data sets of size $n = 250$ objects and we vary the number of clusters k between 2 and 10 and the number of dimensions d both between 2 and 100. The cluster centers are chosen randomly in a $[0, 1]^d$ hypercube and the data points follow a multivariate normal distribution around the center with the covariance matrix $\Sigma = 0.01\mathbf{I}_d$ where $\mathbf{I}_d$ is the $d \times d$ identity matrix. Since we do not evaluate the clustering algorithm, we assume for the experiments that we know the assignment of objects to clusters. We simulate ten data sets with each combination of k and d.

The results are presented in Figure 8. The box plots indicate that while coarse seriation is heavily influenced by k, dissimilarity plots are are only minimally influenced by k or d. These preliminary experiments suggests that dissimilarity plots represent a useful tool for a wide variety of data sets.

6 Conclusion

This paper demonstrates that the two well-known ideas of matrix shading and seriation can be combined into dissimilarity plots, a new and powerful visualization technique that provides many advantages over existing techniques for visualizing partitional clustering. Most notably the new method scales well with the dimensionality of the data and by reordering clusters and objects within clusters can provide a very concise structural representation of the clustering. It was also

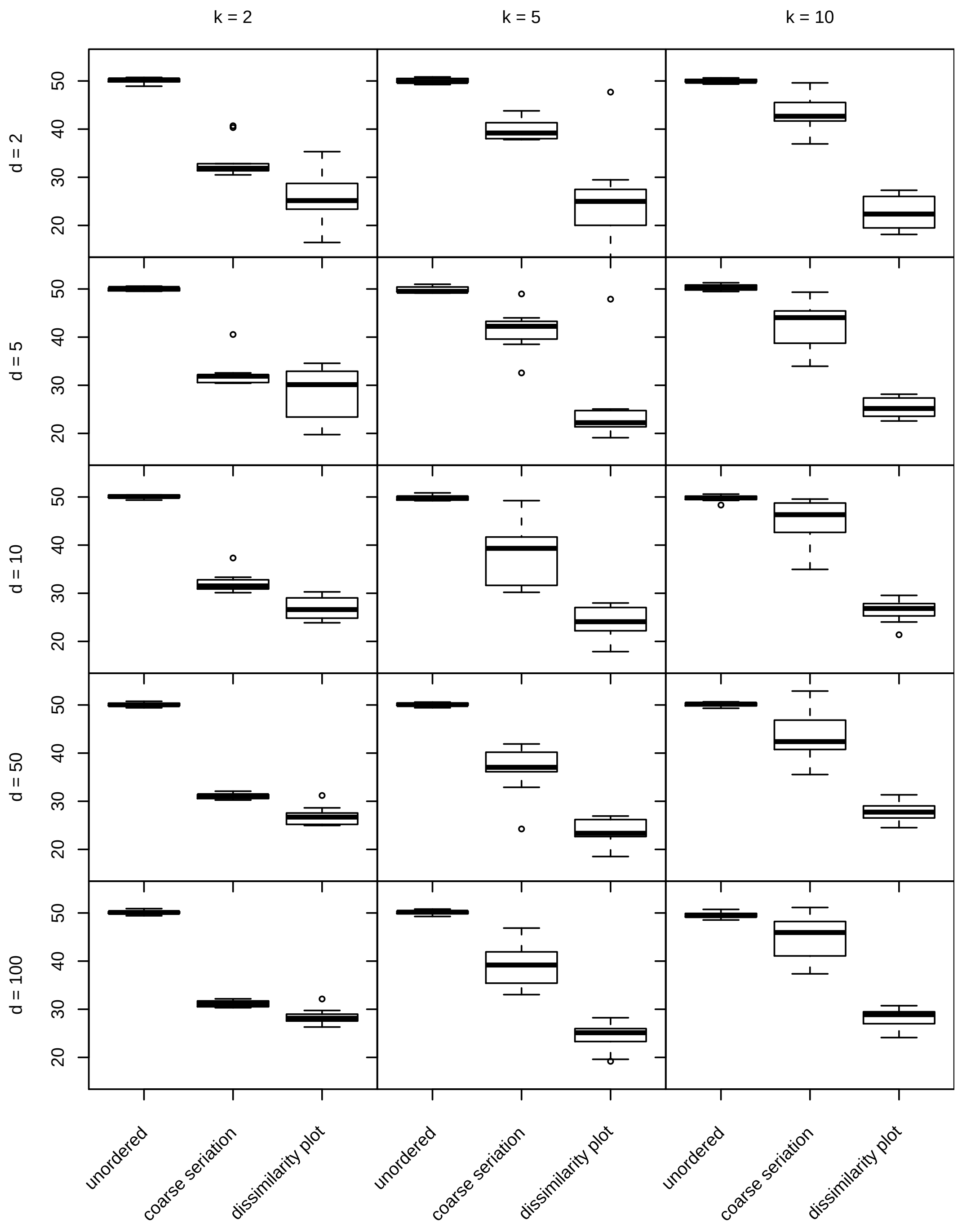


Figure 8: Comparison of the number of anti-Robinson events (in %) left in display for unordered data, coarse seriation and dissimilarity plot for different combinations of k and d.

shown that dissimilarity plots are helpful in spotting the mis-specification of the number of clusters used for partitioning.

An issue with large data sets is display resolution. The resolution of the display used (current mass market displays offer a resolution of up to 1920 × 1200 pixels) restricts the size of the dissimilarity matrix that can be displayed. For larger data sets several methods are possible:

1. Sampling of objects. This reduces the size of the dissimilarity matrix and therefore also speeds up the construction of the dissimilarity plot. However, details are sacrificed.

2. Image downsampling. After the full size reordered dissimilarity matrix is created the size of the image is reduced to fit the display (e.g., by using pixel skipping, pixel averaging or 2D discrete wavelet transformation).

3. Interactive plot. To retain all information in the plot, in the first stage only a plot with dissimilarities between clusters is displayed. Then the user can zoom into individual clusters using the whole available display for only a single cluster.

The first two options are already available for the implementation of dissimilarity plots provided as supplemental material for this paper. Sampling is a built-in function in R and image downsampling in done by R's graphics device. The third option currently needs to be performed manually and a more convenient solution is part of future development.

7 Supplemental Materials

R-package seriation: R-package "seriation" contains the function `dissplot()` which implements dissimilarity plots. (seriation_1.0-2.gz.tar, GNU zipped tar file)

R-code: Complete R-code to reproduce the results in this paper and some additional experiments. (dissplot.R)

References

Applegate, D., Bixby, R., Chvátal, V., and Cook, W. (2006), *Concorde TSP Solver*, School of Industrial and Systems Engineering, Georgia Tech.

Arabie, P. and Hubert, L. J. (1996), "An Overview of Combinatorial Data Analysis," in *Clustering and Classification*, eds. Arabie, P., Hubert, L. J., and Soete, G. D., River Edge, NJ: World Scientific, pp. 5–63.

Asuncion, A. and Newman, D. (2007), "UCI Machine Learning Repository," University of California, Irvine, School of Information and Computer Sciences.

Bar-Joseph, Z., Demaine, E. D., Gifford, D. K., and Jaakkola, T. (2001), "Fast Optimal Leaf Ordering for Hierarchical Clustering," *Bioinformatics*, 17, 22–29.

Bertin, J. (1981), *Graphics and Graphic Information Processing*, Berlin: Walter de Gruyter, translated by William J. Berg and Paul Scott.

Brusco, M., Köhn, H. F., and Stahl, S. (2008), "Heuristic Implementation of Dynamic Programming for Matrix Permutation Problems in Combinatorial Data Analysis," *Psychometrika*, 73, 503–522.

Brusco, M. and Stahl, S. (2005), *Branch-and-bound Applications in Combinatorial Data Analysis*, Springer.

Caraux, G. and Pinloche, S. (2005), "Permutmatrix: A Graphical Environment to Arrange Gene Expression Profiles in Optimal Linear Order," *Bioinformatics*, 21, 1280–1281.

Chen, C.-H. (2002), "Generalized Association Plots: Information Visualization via Iteratively Generated Correlation Matrices," *Statistica Sinica*, 12, 7–29.

Eisen, M. B., Spellman, P. T., Browndagger, P. O., and Botstein, D. (1998), "Cluster Analysis and Display of Genome-wide Expression Patterns," *Proceedings of the National Academy of Science of the United States*, 95, 14863–14868.

Friendly, M. (1994), "Mosaic Displays for Multi-way Contingency Tables," *Journal of the American Statistical Association*, 89, 190–200.

— (2002), "Corrgrams: Exploratory Displays for Correlation Matrices," *The American Statistician*, 56, 316–324.

Gale, N., Halperin, W. C., and Costanzo, C. M. (1984), "Unclassed Matrix Shading and Optimal Ordering in Hierarchical Cluster Analysis," *Journal of Classification*, 1, 75–92.

Gruvaeus, G. and Wainer, H. (1972), "Two Additions to Hierarchical Cluster Analysis," *British Journal of Mathematical and Statistical Psychology*, 25, 200–206.

Gutin, G. and Punnen, A. P. (eds.) (2002), *The Traveling Salesman Problem and Its Variations*, vol. 12 of *Combinatorial Optimization*, Dordrecht: Kluwer.

Hahsler, M., Buchta, C., and Hornik, K. (2010), *seriation: Infrastructure for Seriation*, R package version 1.0-2.

Hahsler, M., Hornik, K., and Buchta, C. (2008), "Getting Things in Order: An Introduction to the R Package seriation," *Journal of Statistical Software*, 25, 1–34.

Hartigan, J. A. (1967), "Representation of Similarity Matrices by Trees," *Journal of the American Statistical Association*, 62, 1140–1158.

Hausdorff, F. (2001), *Set Theory*, American Mathematical Society, 5th ed.

Huband, J. M., Bezdek, J. C., and Hathaway, R. J. (2005), "bigVAT: Visual Assessment of Cluster Tendency for Large Data Sets," *Pattern Recognition*, 38, 1875–1886.

Hubert, L., Arabie, P., and Meulman, J. (1987), *Combinatorial Data Analysis: Optimization by Dynamic Programming*, Society for Industrial Mathematics.

Hubert, L. J. (1974), "Some Applications of Graph Theory and Related Nonmetric Techniques to Problems of Approximate Seriation: The Case of Symmetric Proximity Measures," *British Journal of Mathematical Statistics and Psychology*, 27, 133–153.

Hurley, C. B. (2004), "Clustering Visualizations of Multidimensional Data," *Journal of Computational and Graphical Statistics*, 13, 788–806.

Kaufman, L. and Rousseeuw, P. J. (1990), *Finding Groups in Data: An Introduction to Cluster Analysis*, New York: John Wiley and Sons.

Leisch, F. (2008), "Visualizing Cluster Analysis and Finite Mixture Models," in *Handbook of Data Visualization*, eds. Chen, C., Härdle, W., and Unwin, A., Springer Verlag, Springer Handbooks of Computational Statistics.

Lin, S. and Kernighan, B. (1973), "An Effective Heuristic Algorithm for the Traveling-salesman Problem," *Operations Research*, 21, 498–516.

Ling, R. L. (1973), "A Computer Generated Aid for Cluster Analysis," *Communications of the ACM*, 16, 355–361.

Pison, G., Struyf, A., and Rousseeuw, P. J. (1999), "Displaying a Clustering with CLUSPLOT," *Computational Statistics & Data Analysis*, 30, 381–392.

Robinson, W. S. (1951), "A Method for Chronologically Ordering Archaeological Deposits," *American Antiquity*, 16, 293–301.

Rousseeuw, P. J. (1987), "Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis," *Journal of Computational and Applied Mathematics*, 20, 53–65.

Ruspini, E. H. (1970), "Numerical Methods for Fuzzy Clustering," *Information Science*, 2, 319–350.

Sneath, P. H. A. and Sokal, R. R. (1973), *Numerical Taxonomy*, San Francisco: Freeman and Company.

Strehl, A. and Ghosh, J. (2003), "Relationship-based Clustering and Visualization for High-dimensional Data Mining," *INFORMS Journal on Computing*, 15, 208–230.

Wilkinson, L. and Friendly, M. (2009), "The History of the Cluster Heat Map," *The American Statistician*, 63, 179–184.

Wishart, D. (1999), "ClustanGraphics3: Interactive Graphics for Cluster Analysis," in *Data Analysis and Knowledge Organization: Classification in the Information Age*, eds. Gaul, W. and Locarek-Junge, H., Springer, Studies in Classification, pp. 268–275.

Zeileis, A., Hornik, K., and Murrell, P. (2009), "Escaping RGBland: Selecting Colors for Statistical Graphics," *Computational Statistics & Data Analysis*, 53, 3259–3270.